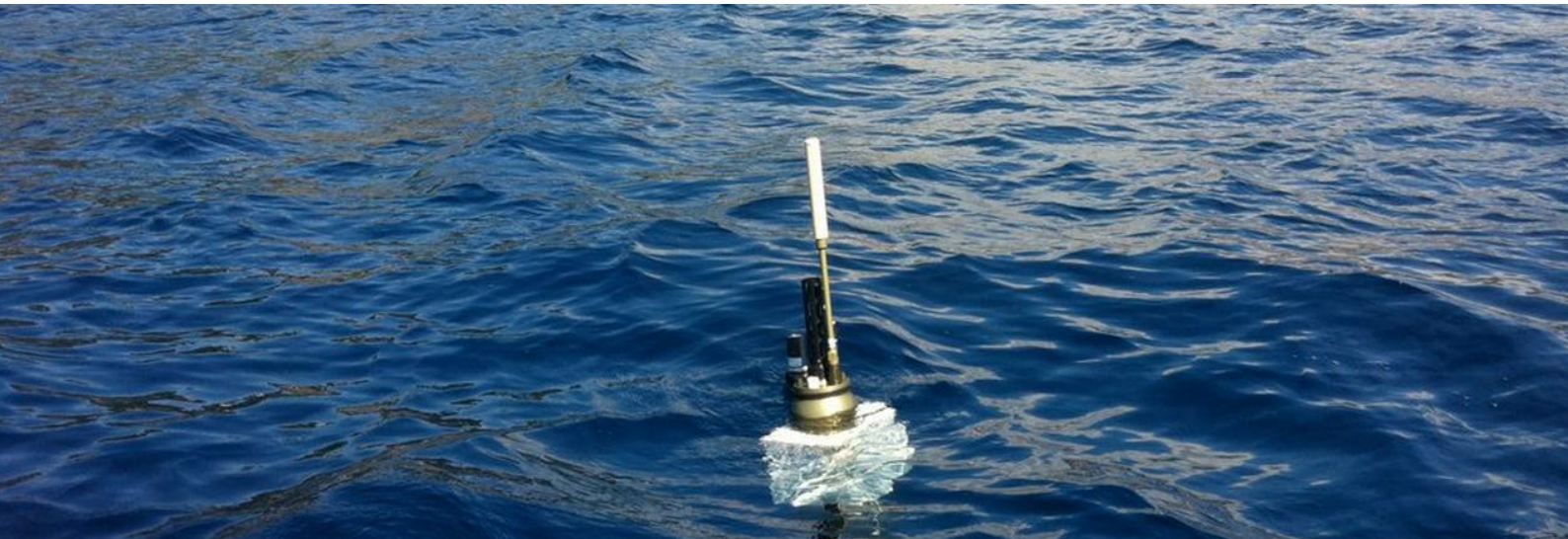


APMT Profiler - File management

AUTOMATED **MULTI-TASK** PROFILER



33-16-046_APMT_File_Management
Revision 1.9 (2020-10-23)

Table of contents

Table of contents.....	2
1. Revision history	5
2. File system	7
2.1 Memory organisation	7
2.2 File types.....	7
2.3 Main memory – Working memory	7
2.4 Extended memory – Storage memory (optional).....	7
3. Configuration files	9
3.1 Local configuration via Bluetooth	9
3.2 Remote configuration via remote commands	9
3.3 Transmission of the configuration file.....	9
4. Technical files	10
4.1 Transmission of technical information.....	10
4.1.1. Self-test files	10
4.1.2. Mission files	10
4.1.3. End-of-life files.....	10
4.2 Technical information available	11
[SYSTEM].....	11
[GPS]	11
[USER]	11
[ACTIVATION]	11
[PROFILE]	11
[DATA]	14
[POWER]	14
[ALARM].....	15
[SENSOR_SUNA]	17
[SENSOR_UVP6].....	17
5. Metadata files.....	19
5.1. Transmission of metadata information.....	19
5.2. Metadata information available.....	19
[PROFILER]	19
[TELECOM].....	19



[HARDWARE]	19
[SENSOR_SBE41].....	20
[SENSOR_DO]	20
[SENSOR_OCR].....	20
[SENSOR_ECO].....	20
[SENSOR_ECOv2]	20
[SENSOR_CROVER]	21
[SENSOR_SBEPH]	21
[SENSOR_SUNA]	21
[SENSOR_UVP6].....	21
[SENSOR_RAMSES]	21
[SENSOR_OPUS]	22
6. Sensor data files	24
6.1. Transmission of sensor data.....	24
6.2. ID tags.....	24
6.2.1. Navigation phase identification.....	24
6.2.2. Processing type identification	25
6.3. Recording in text format	25
6.4. Recording in binary format.....	26
6.4.1. Encoding format	26
6.4.2. File structure.....	26
6.4.3. Timestamping	27
6.5. SBE41 sensor	28
6.5.1. Data encoding.....	28
6.5.2. Standard binary format	28
6.5.3. Extended binary format.....	33
6.6. DO sensor	39
6.7. OCR sensor	44
6.8. ECO sensor.....	49
6.9. ECOv2 sensor.....	54
6.10. CROVER sensor	59
6.11. SBEPH sensor	62
6.12. SUNA sensor	66
6.13. UVP6 sensor (LPM)	70



6.14.	UVP6 sensor (BLACK)	73
6.15.	EXTTRIG sensor	75
6.16.	RAMSES sensor	76
6.18.	OPUS sensor (LIGHT)	79
6.19.	OPUS sensor (BLACK).....	82



1. Revision history

Revision	Release date	Notes	Author
1.0	2017-09-04	Original	C. SCHAEFFER
1.1	2018-02-15	Upgrade of technical information: <ul style="list-style-type: none"> - Adding internal temperature information in [SYSTEM] - Adding pressure activation information in [ACTIVATION] - Adding nke ID information in [USER] - Changing grounding information during navigation phases in [PROFILE] and [ALARM] - Changing feedback information in [ALARM] 	C. SCHAEFFER
1.2	2018-11-19	Upgrade of technical information: <ul style="list-style-type: none"> - Adding pressure reference in "VBatt peak min" tag - Adding USEA identification in [USER] - Adding sensor identification [SENSOR_xxx] - Adding surface sample count in [DATA] - Adding sensor tag in [POWER] - Adding alarms flag in [ALARM] Upgrade of sensor data files: <ul style="list-style-type: none"> - Adding navigation phase [SURFACE] - Adding decimated raw treatment tag (DW) - Adding a 100 dbar offset in SBE41 pressure - Adding DO, OCR, ECO, C-ROVER, SBEPH, SUNA data format 	C. SCHAEFFER
1.3	2019-12-12	Upgrade of technical information: <ul style="list-style-type: none"> - Adding metadata files - Removing redundant information [USER] - Removing redundant information [SENSOR_xxx] - Changing "kB" into "KB" Upgrade of sensor data files <ul style="list-style-type: none"> - Adding UVP6 data formats Update SBE41 *.hex standard format: <ul style="list-style-type: none"> - Adding time stamping for each record Upgrade of technical information: <ul style="list-style-type: none"> - Adding ice capabilities alarms 	C. SCHAEFFER
1.4	2020-01-20	Upgrade of sensor data files <ul style="list-style-type: none"> - Adding EXTTRIG data format 	C. SCHAEFFER
1.5	2020-02-11	Upgrade of sensor data files <ul style="list-style-type: none"> - Adding ECOv2 data format - Adding technical and metadata information related to ECOv2 	C. SCHAEFFER
1.6	2020-03-12	Upgrade of technical information <ul style="list-style-type: none"> - Adding ice braking and perigee in [PROFILE] - Adding alarm flag in [ALARM] 	C. SCHAEFFER
1.7	2020-08-26	Upgrade of technical and sensor data files <ul style="list-style-type: none"> - Adding RAMSES data format 	C. SCHAEFFER
1.8	2020-09-30	Upgrade of metadata file: <ul style="list-style-type: none"> - Changing profiler model format - Adding APMT's SD card information 	C. SCHAEFFER



		<ul style="list-style-type: none">- Adding BATTERY information- Adding SBE41 firmware information	
1.9	2020-10-22	<p>Upgrade of technical and sensor data files</p> <ul style="list-style-type: none">- Adding OPUS data format <p>Upgrade of metadata file:</p> <ul style="list-style-type: none">- Adding PROFILER.ID information- Adding TELECOM.Login information- Adding ECO.Type information- Adding EXTENSION_BOARD information- Adding HULL information	C. SCHAEFFER



2. File system

Float management is based on a specific file system developed by nke. This file system allows information and data to be stored when the float is turned off, together with the mission configuration.

These files can be accessed by an FTP client via the Bluetooth connection.

2.1 Memory organisation

The memory space is divided into different areas intended for different uses. In particular, the following spaces can be found:

- Reserved: restricted system use to store file manipulation and management information
- Working: used for temporary storage (awaiting transmission).
- Storage (optional): used to expand the temporary storage capacity of the workspace

2.2 File types

The system uses various file types, as needed:

- Configuration file for mission and sensor settings
- Sensor data files (raw, processed and transmitted data)
- Technical data files
- Metadata files (for sensor specific identification)
- Execution trace files (mission progress monitoring)
- Specific files (restricted system use)

2.3 Main memory – Working memory

The working memory (fast access) is itself divided into two physical spaces:

- Area for repeated access files: used for files with recurring access (configuration file...)
- Area for temporary files: used mainly for the generation of sensor information and data during the various navigation phases (a process that automatically deletes the oldest files in favour of new ones allows a minimum free memory space to be maintained)

2.4 Extended memory – Storage memory (optional)

When the application requires it, the use of an additional memory (slow access) allows the storage capacity of the system to be increased.



In this case, the files (awaiting transmission) deleted from the working memory are moved to the extended memory. They will then be repatriated to the working memory (for transmission) when the main memory space is sufficient.



3. Configuration files

The mission configuration is saved as a text file organised by sections and tags.

3.1 Local configuration via Bluetooth

During the mission configuration phase performed locally via Bluetooth, this file can be written and/or modified by the user. It can be manually uploaded in full using an FTP client or partially modified with a TELNET command interpreter.

3.2 Remote configuration via remote commands

When the mission is in progress, the file can be modified by sending remote commands via the satellite communication system (RUDICS...).

3.3 Transmission of the configuration file

The system transmits the current configuration file at various stages of the mission:

- At the start of the mission (initial configuration)
- Upon a change of configuration following the application of a remote command or pre-programmed change (script)

The various files generated are named using the format "xxxx_ccc_mm_apmt.ini", with:

- "xxxx": float's hexadecimal serial number (4 characters)
- "ccc": cycle number (3 characters)
- "mm": pattern number (2 characters)



4. Technical files

The various pieces of technical information associated with a profile are saved in a text file organised by sections.

4.1 Transmission of technical information

The system transmits technical files at various stages of the mission:

- During system verification prior to the mission (self-test)
- During the mission at the end of each pattern
- At end of life

4.1.1. Self-test files

The files generated are named using the format "xxxx_ccc_autotest_nnnnn.txt", with:

- "xxxx": float's hexadecimal serial number (4 characters)
- "ccc": cycle number (3 characters)
- "nnnnn": identification of self-test file number associated with the cycle (5 characters)

4.1.2. Mission files

The files generated are named using the format "xxxx_ccc_mm_technical.txt":

- "xxxx": float's hexadecimal serial number (4 characters)
- "ccc": cycle number (3 characters)
- "mm": pattern number (2 characters)

Note: A pattern numbered "00" corresponds to a pre-mission file.

4.1.3. End-of-life files

The various files generated are named using the format "xxxx_ccc_mm_default_nnnnn.txt", with:

- "xxxx": float's hexadecimal serial number (4 characters)
- "ccc": cycle number (3 characters)
- "mm": pattern number (2 characters)
- "nnnnn": identification of end-of-life file number associated with the cycle/pattern pair (5 characters)



4.2 Technical information available

The data is organised in sections. The various sections and fields that compose them are optional and may not be present in the files.

[SYSTEM]

Information	Format
External pressure offset	Pe offset=xx.xx dbar
Internal pressure	Pi=xxx.x mbar
No-load battery voltage	Vbatt=xx.x V
Battery voltage under load (min.)	Vbatt peak min=xx.x V (xx dbar)
External temperature (air)	Te air=xx.x degC
External pressure (air)	Pe air=xx.x dbar
Internal temperature	Ti=xx.x degC

[GPS]

Information	Format
Time stamping, position (*) and clock drift (**)	UTC=yy-mm-dd hh:mm:ss Lat=ddmm.mmmmmh Long=dddmm.mmmmmh Clock drift=xx.xxx s

(*) Geographic coordinates are coded in DM format (degrees, minutes and decimal minutes – NMEA standard). For example: “Lat=4747.846N Long=00317.160W” represents 47°47.846’N 3°17.160’W coordinates.

(**) Drift = Float time - GPS time. The offset is reset to 0 after every GPS retiming.

[USER]

Information	Format
nke ID	WC=xxxxxxxxxx

[ACTIVATION]

Information	Format
Pressure activation start	UTC=yy-mm-dd hh:mm:ss Detection start
Pressure activation stop	UTC=yy-mm-dd hh:mm:ss Detection stop

[PROFILE]

Information	Format
Emergence reduction: <ul style="list-style-type: none"> - Start date/time - Oil volume transferred - Nb. solenoid valve actions 	UTC=yy-mm-dd hh:mm:ss Flotation=xxx.x cm3 (xx)
First stabilization <ul style="list-style-type: none"> - Date/time - Depth 	UTC=yy-mm-dd hh:mm:ss First stabilization=xxx dbar
Park descent: <ul style="list-style-type: none"> - Start date/time - Oil volume transferred - Nb. solenoid valve actions 	UTC=yy-mm-dd hh:mm:ss Descent=xxx.x cm3 (xx)
Grounding park descent:	UTC=yy-mm-dd hh:mm:ss Grounding Descent=xxx dbar



<ul style="list-style-type: none"> - Start date/time - Grounding pressure 	
Grounding park descent: <ul style="list-style-type: none"> - End date/time - Oil volume transferred 	UTC=yy-mm-dd hh:mm:ss Grounding Descent escape=xxx.x cm3
Park drift: <ul style="list-style-type: none"> - Start date/time - Min. depth - Max. depth - Nb. solenoid valve actions - Nb. pump actions - Nb. set point inputs - Nb. set point outputs 	UTC=yy-mm-dd hh:mm:ss Park=xxx/xxx dbar (xx/xx) stability=x/x
Stabilized park drift: <ul style="list-style-type: none"> - Stabilization date/time - Stabilization depth 	UTC=yy-mm-dd hh:mm:ss Park stabilization=xxx dbar
Grounding park drift: <ul style="list-style-type: none"> - Start date/time - Grounding number (1-5) - Grounding pressure 	UTC=yy-mm-dd hh:mm:ss Grounding Park x=xxx dbar
Grounding park drift: <ul style="list-style-type: none"> - End date/time - Grounding number (1-5) - Oil volume transferred 	UTC=yy-mm-dd hh:mm:ss Grounding Park x escape=xxx.x cm3
Measurement descent: <ul style="list-style-type: none"> - Start date/time - Oil volume transferred - Nb. solenoid valve actions 	UTC=yy-mm-dd hh:mm:ss Deep profile=xxx.x cm3 (xx)
Grounding measurement descent: <ul style="list-style-type: none"> - Start date/time - Grounding pressure 	UTC=yy-mm-dd hh:mm:ss Grounding Deep profile=xxx dbar
Grounding measurement descent: <ul style="list-style-type: none"> - End date/time - Oil volume transferred 	UTC=yy-mm-dd hh:mm:ss Grounding Deep profile escape=xxx.x cm3
Measurement drift: <ul style="list-style-type: none"> - Start date/time - Min. depth - Max. depth - Nb. solenoid valve actions - Nb. pump actions - Nb. set point inputs - Nb. set point outputs 	UTC=yy-mm-dd hh:mm:ss Short Park=xxx/xxx dbar (xx/xx) stability=x/x
Grounding short park drift: <ul style="list-style-type: none"> - Start date/time - Grounding number (1-5) - Grounding pressure 	UTC=yy-mm-dd hh:mm:ss Grounding Short park x=xxx dbar
Grounding short park drift: <ul style="list-style-type: none"> - End date/time - Grounding number (1-5) - Oil volume transferred 	UTC=yy-mm-dd hh:mm:ss Grounding Short park x escape=xxx.x cm3
Ascent (standard):	UTC=yy-mm-dd hh:mm:ss Ascent=xxx.x cm3 (xx/xx) from xxx



<ul style="list-style-type: none"> - Start date/time - Oil volume transferred - Nb. pump actions (total) - Nb. actions for take-off - Maximum depth 	dbar
Ascent (slow): <ul style="list-style-type: none"> - Start date/time - Oil volume transferred - Nb. pump actions 	UTC=yy-mm-dd hh:mm:ss Ascent (slowly)=xxx.x cm3 (xx)
Ascent (resume): <ul style="list-style-type: none"> - Start date/time - Oil volume transferred - Nb. pump actions 	UTC=yy-mm-dd hh:mm:ss Ascent (resume)=xxx.x cm3 (xx)
Ascent (end) <ul style="list-style-type: none"> - End date/time 	UTC=yy-mm-dd hh:mm:ss Ascent end
Ice abort: <ul style="list-style-type: none"> - Abort date/time - Oil volume transferred - Abort pressure 	UTC=yy-mm-dd hh:mm:ss Abort=xxx.x cm3 at xxx dbar
Ice perigee: <ul style="list-style-type: none"> - Perigee date/time - Perigee pressure 	UTC=yy-mm-dd hh:mm:ss Perigee=xxx dbar
Surface <ul style="list-style-type: none"> - Start date/time 	UTC=yy-mm-dd hh:mm:ss Surface
Hanging: <ul style="list-style-type: none"> - Start date/time - Hanging pressure 	UTC=yy-mm-dd hh:mm:ss Hanging=xxx dbar
Hanging: <ul style="list-style-type: none"> - End date/time 	UTC=yy-mm-dd hh:mm:ss Hanging escape
Park drift (ice): <ul style="list-style-type: none"> - Start date/time - Min. depth - Max. depth - Nb. solenoid valve actions - Nb. pump actions - Nb. set point inputs - Nb. set point outputs 	UTC=yy-mm-dd hh:mm:ss Ice Park=xxx/xxx dbar (xx/xx) stability=x/x
Stabilized park drift (ice): <ul style="list-style-type: none"> - Stabilization date/time - Stabilization depth 	UTC=yy-mm-dd hh:mm:ss Ice Park stabilization=xxx dbar
Pressure switch activation	UTC=yy-mm-dd hh:mm:ss Pressure switch activation
Emergency ascent <ul style="list-style-type: none"> - Start date/time 	UTC=yy-mm-dd hh:mm:ss Emergency ascent

Typical rate of descent can be calculated with: "Park drift" UTC - "Emergence reduction" UTC

Typical rate of ascent can be calculated with: "Ascent (end)" UTC – "Ascent (standard)" UTC



[DATA]

Information	Format
Data transmission (*): <ul style="list-style-type: none"> - Total data size - Nb. files - Average baud rate - Nb. sessions 	Upload=xx.x KB of x file(s) at x.x KB/min in x session(s)
Remote command reception: <ul style="list-style-type: none"> - Nb. accepted - Nb. refused - Nb. unknown 	Download=command file (x accepted, x refused, x unknown)
Payload configuration download	Download=payload file
Script download	Download=script file
Number of files associated with the pattern (**)	Pattern=x files
Number of points per sensor: <ul style="list-style-type: none"> - Sensor name [SBE41, DO, OCR, ECO, ECOv2, CROVER, SBEPH, SUNA, UVP6-LPM, UVP6-TX1, UVP6-TX2, UVP6-BLK, EXT-TRIG, RAMSES, OPUS-LGT, OPUS-BLK] - Nb. park descent - Nb. park drift - Nb. measurement descent - Nb. measurement drift - Nb. ascent - Nb. surface - Nb. subsurface (***) 	xxxxx=x/x/x/x/x/x points or xxxxx=x/x/x/x/x/x/x points (***)

(*) Information from the previous transmission session.

(**) The following files are not taken into account: technical for self-test or pre-mission, configuration nor command/script acknowledgement. The number of files is expressed prior to the files being broken down for transmission.

(***) SBE41 only

[POWER]

Information	Format
Pattern duration	Pattern=x min
Processing/standby ratio	Treatment=xx %
Cumulated hydraulic activations: <ul style="list-style-type: none"> - Solenoid valve - Pump 	EV/Pump=xxx/xxx cs
Cumulated modem activations (*)	Transmission=xxx min
Cumulated GPS activations	GPS=xxx s
Cumulated sensor activations: <ul style="list-style-type: none"> - Sensor name [SBE41, DO, OCR, ECO, ECOv2, CROVER, SBEPH, SUNA, 	xxxx=xxx min



UVP6, EXT-TRIG, RAMSES, OPUS-LGT, OPUS-BLK] - Duration	
--	--

(*) Information from the previous transmission session.

[ALARM]

Information	Format
Deployment	
Start up	Power-on
Invalid configuration	Bad configuration
Excessively heavy float	Flotation (heavy)
Excessively light float	Flotation (light)
Self-test failure: - Source(s) xxx among "FRAM, FLASH, RTC, Vbatt, Pi, Pe, SBE41-Cutoff, SBE41-Offset, GPS, Payload, USEA, Sensor(xxx), Transmitter"	Autotest fail=xxx, ...
State	
No-load battery voltage low	Vbatt low
Battery voltage under load low	Vbatt peak low
Low external pressure	Pe low (xxx dbar)
High external pressure	Pe high (xxx dbar)
External pressure fault	Pe default
External breaking pressure	Pe broken
Gear skip	Pe SR high
High internal pressure	Pi high
Presence of water	Water inside
Navigation	
Grounding during park descent	Grounding Descent (xxx dbar)
Grounding during park drift	Grounding Park x (xxx dbar)
Grounding during measurement descent	Grounding Deep profile (xxx dbar)
Grounding during short park drift	Grounding Short park x (xxx dbar)
Hanging during ascent	Hanging (xxx dbar)
Braking during descent	Braking
Operating errors	
System fault	System
Payload board fault	Payload
GPS fault	GPS
Hydraulic fault	Hydraulic
ADC fault	ADC
File fault	File (skip)
RTC fault	RTC
Pressure switch fault	Pressure switch
Ice capabilities	



Ice avoidance (ISA detection)	Ice (ISA)
Ice avoidance (collision detection)	Ice (collision)
Ice avoidance (aborting profile)	Ice (abort)
Ice avoidance (cover detection)	Ice (cover)
Ice avoidance (no surface period)	Ice (period)
Mode switch	
End of life: <ul style="list-style-type: none"> - Source xxx among "Pi high, Pe broken, Pe high, Vbatt low, Vbatt peak low, Water inside, Flotation (heavy), Flotation (light)" 	End of life (xxx)
Rescue procedure	Rescue
Feedback: <ul style="list-style-type: none"> - Type xxx among "Early profile, Early surface, Abort profile, Abort cycle 0, Abort cycle 1, Go to deep, Standard speed, Lower speed" - Nb. accepted - Nb. refused 	Feedback=xxx (x accepted, x refused) Feedback=xxx (x accepted, x refused) ...
Sensors	
Bad value in data frame: <ul style="list-style-type: none"> - Sensor name [SBE41, DO, OCR, ECO, ECOv2, CROVER, SBEPH, SUNA, UVP6, RAMSES, OPUS] - Error count 	xxx value (xxx)
No reply/framing error: <ul style="list-style-type: none"> - Sensor name [SBE41, DO, OCR, ECO, ECOv2, CROVER, SBEPH, SUNA, UVP6, RAMSES, OPUS] - Error count 	xxx default (xxx)
Repetitive default: <ul style="list-style-type: none"> - Sensor name [SBE41, DO, OCR, ECO, ECOv2, CROVER, SBEPH, SUNA, UVP6, RAMSES, OPUS] 	xxx broken
Data size: <ul style="list-style-type: none"> - Sensor name [SBE41, DO, OCR, ECO, ECOv2, CROVER, SBEPH, SUNA, UVP6-LPM, UVP6-TX1, UVP6-TX2, UVP6-BLK, EXT-TRIG, RAMSES, OPUS-LGT, OPUS-BLK] - Data size 	xxx size (xxx.x KB)



[SENSOR_SUNA]

Information	Format
Counters: <ul style="list-style-type: none">- Sample counter- Power cycle counter- Error counter	Counters=892340/852111/16
Power supply: <ul style="list-style-type: none">- Voltage- Intensity	Power supply=9.53 V/0.663 A

[SENSOR_UVP6]

Information	Format
SD memory free space	Available space=xxx.x GB



Example:**[USER]**

WC=85E386B7C7

[SYSTEM]

Pe offset=0.00 dbar

Pe air=0.00 dbar

Pi=750.3 mbar

Vbatt=10.7 V

Vbatt peak min=10.0 V (0 dbar)

[GPS]

UTC=18-11-20 12:17:24 Lat=4310.35000S Long=00555.02500W Clock drift=+0.000 s

[PROFILE]

UTC=18-11-20 10:12:22 Flotation=10.7 cm3 (1)

UTC=18-11-20 10:34:03 First stabilization=9 dbar

UTC=18-11-20 10:23:45 Descent=12.2 cm3 (4)

UTC=18-11-20 11:33:06 Park=68/68 dbar (0/0) stability=0/0

UTC=18-11-20 11:33:08 Deep profile=0.0 cm3 (0)

UTC=18-11-20 11:34:06 Short Park=68/68 dbar (0/0) stability=0/0

UTC=18-11-20 11:34:07 Ascent=24.6 cm3 (12/4) from 69 dbar

UTC=18-11-20 12:00:25 Ascent end

UTC=18-11-20 12:10:25 Surface

[DATA]

Upload=4.8 KB of 3 file(s) at 10.2 KB/min in 1 session(s)

Pattern=6 files

SBE41=47/0/0/0/30/6/1 points

OCR=95/0/0/0/33/0 points

ECO=86/0/0/0/30/0 points

SBE41=58/0/0/0/32/0 points

SUNA=7/0/0/0/4/0 points

[POWER]

Pattern=126 min

Treatment=1 %

EV/Pump=337/382 cs

SBE41=115 min

Transmission=2 min

GPS=260 s

OCR=6 min

ECO=6 min

SBE41=6 min

SUNA=1 min

[SENSOR_SUNA]

Counters=892340/852111/16

Power supply=9.53 V/0.663 A



5. Metadata files

5.1. Transmission of metadata information

Metadata files are transmitted once at the beginning of the mission.

The files generated are named using the format "xxxx_ccc_mm_metadata.xml", with:

- "xxxx": float's hexadecimal serial number (4 characters)
- "ccc": cycle number (3 characters)
- "mm": pattern number (2 characters)

5.2. Metadata information available

Metadata information are organised in XML format.

[PROFILER]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Identifier - Model 	<PROFILER SN="P5XXXX-XXXXXXX" ID="0XXXXX" Model="PROVOR_V"/>

[TELECOM]

Information	Format
Identification: <ul style="list-style-type: none"> - Type - SIM card ID - Account login 	<TELECOM Type="IRIDIUM" CID="xxxxxxxxxxxxx" Login="xxxxx"/>

[HARDWARE]

Information	Format
Control board identification: <ul style="list-style-type: none"> - Model - Firmware version - SD card status 	<CONTROL_BOARD Model="APMT" Firmware="x.xx.xxx" SDCard="Installed/Not installed"/>
Measure board identification: <ul style="list-style-type: none"> - Model - Firmware version 	<MEASURE_BOARD Model="USEA" Firmware="x.xx.xxx"/>
Extension board identification: <ul style="list-style-type: none"> - Model 	<EXTENSION_BOARD Model="EXTEND"/>
Battery configuration: <ul style="list-style-type: none"> - Pack number "x" information - Type - Voltage - capacity 	<PACK_x Type="Lithium" Voltage="10.8 V" Capacity="xxx Ah"/>
Hull configuration:	<HULL Model="Type x"/>



- Model	
---------	--

[SENSOR_SBE41]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Model 	<SENSOR SN="XXXXX" Model="SBE41-CP"/>
Sensor board identification: <ul style="list-style-type: none"> - Firmware version 	<SBE41_BOARD Firmware="x.x.x"/>
Pressure sensor Identification: <ul style="list-style-type: none"> - Serial number 	<SENSOR_PRESSURE SN="XXXXX"/>

[SENSOR_DO]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Model 	<SENSOR SN="XXXXX" Model="DOxxx"/>
Phase correction: <ul style="list-style-type: none"> - Coefficient "c0" 	<PHASE_COEFF c0="xxxx"/>
SVU foil correction: <ul style="list-style-type: none"> - Coefficient "c0" - ... - Coefficient "c6" 	<SVU_FOIL_COEFF c0="xxxx" c1="xxxx" ... c6="xxxx"/>

[SENSOR_OCR]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Model 	<SENSOR SN="XXXXX" Model="OCRxxx"/>
Channel correction: <ul style="list-style-type: none"> - Index [4,7 or 14] - Coefficient "a0" - Coefficient "a1" - Coefficient "im" 	<CHANNEL_xx a0="x.xxxx" a1="x.xxxx" im="x.xxxx"/>

[SENSOR_ECO]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Model - Type 	<SENSOR SN="XXXXX" Model="ECOxxx" Type="xxxxxx"/>
Channel correction: <ul style="list-style-type: none"> - Index [1-3] - Scale factor "sf" - Dark count "dc" 	<CHANNEL_xx sf="x.xxxx" dc="xxxxxxx"/>

[SENSOR_ECOv2]

Information	Format
Identification:	<SENSOR SN="XXXXX" Model="ECOxxx"/>



<ul style="list-style-type: none"> - Serial number - Model 	
Channel correction: <ul style="list-style-type: none"> - Index [1-3] - Scale factor "sf" - Dark count "dc" 	<CHANNEL_xx sf="x.xxxx" dc="xxxxxxx"/>

[SENSOR_CROVER]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number 	<SENSOR SN="XXXXX"/>
Path length: <ul style="list-style-type: none"> - X value "pth" (cm) 	<PATH_LENGTH pth="xx.xx"/>
Calibration: <ul style="list-style-type: none"> - CSCcal value "cln" 	<CALIBRATION cln="xxxxx"/>

[SENSOR_SBEPH]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number 	<SENSOR SN="XXXXX"/>

[SENSOR_SUNA]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Model - Output pixel begin - Output pixel end 	<SENSOR SN="XXXXX" Model="SUNA-V2" Spectrum="Output pixels xx-xx"/>
Sensor board identification: <ul style="list-style-type: none"> - Firmware version 	<SUNA_BOARD Firmware="x.x.xx"/>
Spectrometer: <ul style="list-style-type: none"> - Integration time "spintper" 	<SPECTROMETER spintper="xxx"/>

[SENSOR_UVP6]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number - Model 	<SENSOR SN="XXXXXXXXXXXX" Model="UVP6-LP"/>
HW configuration	<HW_CONF frame="xxxxxxxxxx">
ACQ configuration: <ul style="list-style-type: none"> - Index [1-10] 	<ACQ_CONF_xx frame="xxxxxxxxxx">
TAXO configuration	<TAXO_CONF frame="xxxxxxxxxx">

[SENSOR_RAMSES]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number 	<SENSOR SN="XXXXX"/>
Sensor board identification: <ul style="list-style-type: none"> - Firmware version 	<RAMSES_BOARD Firmware="x.x.xx"/>



Spectrum: <ul style="list-style-type: none"> - Wavelengths number - Wavelengths 	<SPECTRUM Length="xx" Wavelengths="xx.xxx,[...],xx.xxx"/>
---	---

[SENSOR_OPUS]

Information	Format
Identification: <ul style="list-style-type: none"> - Serial number 	<SENSOR SN="XXXXX"/>
Sensor board identification: <ul style="list-style-type: none"> - Firmware version 	<OPUS_BOARD Firmware="x.x.xx"/>
Spectrum: <ul style="list-style-type: none"> - Wavelengths number - Wavelengths 	<SPECTRUM Length="xxx" Wavelengths="xx.xxx,[...],xx.xxx"/>
Lamp Identification: <ul style="list-style-type: none"> - Serial number 	<SENSOR_LAMP SN="xxxxx"/>
Spectrometer: <ul style="list-style-type: none"> - Optical path size 	<SPECTROMETER Path="xx mm"/>
Waterbase reference spectrum: <ul style="list-style-type: none"> - Wavelengths number - Intensities 	<WATERBASE Length="xxx" Intensities="xx.x,[...],xx.x"/>



Example:

```

<FLOAT>
<PROFILER SN="AABBCC-DDEEFF" Model="PROVOR-V"/>
<TELECOM Type="IRIDIUM" CID="8988169224001045237"/>
<HARDWARE>
  <CONTROL_BOARD Model="APMT" Firmware="1.07.019"/>
  <MEASURE_BOARD Model="USEA" Firmware="1.00.019"/>
</HARDWARE>
<SENSORS>
  <SENSOR_UVP6>
    <SENSOR SN="000001HF" Model="UVP6-LP"/>
    <HW_CONF frame="000001HF,0,ACQ_NKE_00h,0,001,1,150,250,,999.000,393819,10000,2,192.14"/>
    <ACQ_CONF_01 frame="ACQ_NKE_00H,0,1.000,1,1,0,0,1,1,10,0,500,1.0,10,10,0,1000,0,40,1e"/>
    <ACQ_CONF_02 frame="ACQ_NKE_00L,0,1.000,1,1,0,0,1,1,10,0,500,1.0,100,10,0,1000,0,40,1e"/>
    <ACQ_CONF_03 frame="ACQ_NKE_01H,0,1.000,1,1,0,0,1,1,10,0,500,1.0,10,10,0,1000,0,40,1e"/>
    <ACQ_CONF_04 frame="ACQ_NKE_01L,0,1.000,1,1,0,0,1,1,10,0,500,1.0,100,10,0,1000,0,40,1e"/>
    <ACQ_CONF_05 frame="ACQ_NKE_20H,0,1.000,1,1,0,0,1,1,10,2,500,1.0,10,10,0,1000,0,40,1e"/>
    <ACQ_CONF_06 frame="ACQ_NKE_20L,0,1.000,1,1,0,0,1,1,10,2,500,1.0,100,10,0,1000,0,40,1e"/>
    <ACQ_CONF_07 frame="ACQ_NKE_21H,0,1.000,1,1,0,0,1,1,10,2,500,1.0,10,10,0,1000,0,40,1e"/>
    <ACQ_CONF_08 frame="ACQ_NKE_21L,0,1.000,1,1,0,0,1,1,10,2,500,1.0,100,10,0,1000,0,40,1e"/>
    <ACQ_CONF_09 frame="ACQ_NKE_CUST_1,0,1.000,1,1,0,0,1,1,10,2,500,1.0,15,10,0,1000,0,40,1e"/>
    <ACQ_CONF_10 frame="ACQ_NKE_CUST_2,0,1.000,1,1,0,0,1,1,10,1,500,1.0,30,10,0,1000,0,40,1e"/>
  </SENSOR_UVP6>
  <SENSOR_DO>
    <SENSOR SN="03014" Model="DO4330"/>
    <PHASE_COEFF c0="9.500000e-02"/>
    <SVU_FOIL_COEFF c0="2.749372e-03" c1="1.217100e-04" c2="2.062590e-06" c3="1.655341e+00"/>
  </SENSOR_DO>
  <SENSOR_OCR>
    <SENSOR SN="00184" Model="OCR504"/>
    <CHANNEL_01 a0="2.14766538910e+09" a1="1.63699685504e-07" im="1.161e+00"/>
    <CHANNEL_02 a0="2.14754852610e+09" a1="2.00131764196e-07" im="1.368e+00"/>
    <CHANNEL_03 a0="2.14742382370e+09" a1="2.04019168366e-07" im="1.365e+00"/>
    <CHANNEL_04 a0="2.14745155870e+09" a1="3.03210835119e-06" im="1.359e+00"/>
  </SENSOR_OCR>
  <SENSOR_ECO>
    <SENSOR SN="02345" Model="ECO3"/>
    <CHANNEL_01 sf="7.200e-03" dc="47"/>
    <CHANNEL_02 sf="2.029e-06" dc="43"/>
    <CHANNEL_03 sf="8.720e-02" dc="30"/>
  </SENSOR_ECO>
  <SENSOR_SUNA>
    <SENSOR SN="00555" Model="SUNA-V2" Spectrum="Output pixels 38-77"/>
    <SUNA_BOARD Firmware="2.2.13"/>
    <SPECTROMETER spintper="400"/>
  </SENSOR_SUNA>
  <SENSOR_SBE41>
    <SENSOR SN="08959" Model="SBE41-CP"/>
    <SENSOR_PRESSURE SN="004978241"/>
  </SENSOR_SBE41>
</SENSORS>
</FLOAT>

```



6. Sensor data files

The system can generate sensor data files under three formats:

- Text format (*.csv)
- Standard binary format (*.hex)
- Extended binary format (*.hex)

6.1. Transmission of sensor data

The various sensor data files generated are named using the format "xxxx_ccc_mm_ssssss.ext", with:

- "xxxx": float's hexadecimal serial number (4 characters)
- "ccc": cycle number (3 characters)
- "mm": pattern number (2 characters)
- "sssss": sensor ID (N characters, e.g. "sbe41")
- "ext": file extension

All data from a sensor are saved in a unique data file associated to the cycle/pattern.

6.2. ID tags

All files are organised in sections using ID tags for the phase and processing type.

6.2.1. Navigation phase identification

At every change of navigation phase, an ID string is inserted. The various ASCII strings are as follows:

- **[DESCENT]**: Data from the descent phase, from the surface to the drift depth
- **[PARK]**: Data from the drift phase
- **[DEEP_PROFILE]**: Data from the descent phase, from the drift depth to the measurement depth
- **[SHORT_PARK]**: Data from the drift phase, from the drift depth to the measurement depth
- **[ASCENT]**: Data from the ascent phase
- **[SURFACE]**: Data from the surface phase



6.2.2. Processing type identification

At every change of phase or processing area, an ID string for the processing type is inserted. The various ASCII strings are as follows:

- **(RW)**: For raw data
- **(DW)**: For decimated raw data
- **(AM)**: For arithmetic mean
- **(SD)**: For standard deviation
- **(MD)**: For median
- **(SS)**: For subsurface point

Several processing types can follow one another. The combinations are as follows:

- (RW)
- (DW)
- (AM)
- (AM)(SD)
- (AM)(MD)
- (AM)(SD)(MD)
- (SS)

6.3. Recording in text format

Spreadsheet format with the possibility of choosing the decimal separator, the column separator as well as the timestamping format can be selected.

[DESCENT]									
(AM) (SD) (MD)									
2015-06-15	08:34:51	;	4.90;	17.4640;	35.798;	0.0000;	0.0000;	4.90;	17.4640;35.798
2015-06-15	08:34:51	;	5.50;	17.4600;	35.797;	0.0030;	0.0000;	5.50;	17.4600;35.797
2015-06-15	08:34:51	;	6.40;	17.4530;	35.797;	0.0030;	0.0000;	6.40;	17.4530;35.797
2015-06-15	08:34:51	;	7.40;	17.4470;	35.796;	0.0030;	0.0000;	7.40;	17.4500;35.797
2015-06-15	08:34:51	;	8.50;	17.4380;	35.796;	0.0030;	0.0000;	8.50;	17.4360;35.796
2015-06-15	08:34:51	;	9.50;	17.4320;	35.796;	0.0030;	0.0000;	9.50;	17.4290;35.796



6.4. Recording in binary format

6.4.1. Encoding format

The data is saved in "little endian" format with byte alignment objects:

- **BYTE** - char (1 byte)
- **USHORTINT** - unsigned short integer (2 bytes)
- **SHORTINT** - short integer (2 bytes)
- **LONGINT** - long integer (4 bytes)
- **EPOCH** – absolute timestamping (4 bytes) - Unix Epoch format - January 1st, 1970 at 0:00
- **FLOAT** - floating (4 bytes - float) – IEEE-754 format

6.4.2. File structure

Each file begins with an encoding identifying byte:

- 0x01: SBE41 sensor - Extended format
- 0x02: SBE41 sensor - Standard format
- 0x03: DO sensor
- 0x04: OCR504 sensor (4 channels)
- 0x05: OCR507 sensor (7 channels)
- 0x06: OCR507-IR sensor (14 channels)
- 0x07: ECO1 sensor (1 channel)
- 0x08: ECO2 sensor (2 channels)
- 0x09: ECO3 sensor (3 channels)
- 0x0A: CROVER sensor
- 0x0B: SBEPH sensor
- 0x0C: SUNA sensor (45 outputs spectrum)
- 0x0D: SUNA sensor (90 outputs spectrum)
- 0x0E: UVP6 sensor (LPM data)
- 0x0F: UVP6 sensor (TAXO 1 data)
- 0x10: UVP6 sensor (TAXO 2 data)
- 0x11: UVP6 sensor (BLACK data)
- 0x12: ECO1v2 sensor (1 channel)
- 0x13: ECO2v2 sensor (2 channels)
- 0x14: ECO3v2 sensor (3 channels)
- 0x15: ECO4v2 sensor (4 channels)
- 0x16: EXTTRIG sensor
- 0x17: RAMSES sensor
- 0x18: OPUS sensor (LIGHT data)
- 0x19: OPUS sensor (BLACK data)

This is followed by groups of recordings identified by the tags.



Example: SBE41 standard format file, descent phase

00000000	01	5b	44	45	53	43	45	4e	54	5d	28	44	57	29	cb	65	[DESCENT] (DW) Ēe
00000010	e4	5b	00	00	12	04	c7	57	d6	8b	36	55	00	1e	04	c0	ä [...ÇWÖ< 6U...À
00000020	57	d6	8b	45	56	00	29	04	b2	57	d5	8b	13	80	00	36	WÖ< EV.) . *WÖ< .€ . 6
00000030	04	ab	57	d4	8b	62	80	00	41	04	a4	57	d4	8b	61	80	.«WÖ< b€ . A. »WÖ< a€
00000040	00	4f	04	9d	57	d4	8b	50	80	00	5d	04	8e	57	d2	8b	. O. WÖ< P€ .] . ŽWÖ<
00000050	38	81	00	68	04	87	57	d1	8b	67	aa	00	74	04	80	57	8 . h. #WŃ< g² . t. €W

Note: In Iridium RUDICS transmission, files can be supplemented by padding bytes (0x1A). These bytes must not be decoded. Each file can end with a sequence of 0 to 1023 padding bytes.

6.4.3. Timestamping

Records can be timestamped with absolute timestamps (Unix Epoch format) or relative timestamps since a reference date.

Absolute timestamping is used for raw data records in parking phase (where the delay between records can be greater than hours). Data is encoded as follows:

- **Timestamp** absolute encoding (EPOCH)

Relative timestamping is used for the others records. The reference date is saved after the navigation phase tag. The following records use an offset (count in seconds) from this reference time. Data is encoded as follows:

- **Timestamp** offset in seconds (unsigned integer) – code = value - reference time

Example: Relative timestamping encoding

00000000	01	5b	44	45	53	43	45	4e	54	5d	28	44	57	29	cb	65	[DESCENT] (DW) Ēe
00000010	e4	5b	00	00	12	04	c7	57	d6	8b	36	55	00	1e	04	c0	ä [...ÇWÖ< 6U...À
00000020	57	d6	8b	45	56	00	29	04	b2	57	d5	8b	13	80	00	36	WÖ< EV.) . *WÖ< .€ . 6
00000030	04	ab	57	d4	8b	62	80	00	41	04	a4	57	d4	8b	61	80	.«WÖ< b€ . A. »WÖ< a€
00000040	00	4f	04	9d	57	d4	8b	50	80	00	5d	04	8e	57	d2	8b	. O. WÖ< P€ .] . ŽWÖ<
00000050	38	81	00	68	04	87	57	d1	8b	67	aa	00	74	04	80	57	8 . h. #WŃ< g² . t. €W

Timestamp = 1541694923 (0x5BE465CB) + 0 sec (0x0000) => 2018-11-08 16:35:23



6.5.SBE41 sensor

6.5.1. Data encoding

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Temperature** in m°C and +5.0°C offset (unsigned integer) – code = (value + 5.0) * 1000.0
- **Salinity** in mpsu (unsigned integer) – code = value * 1000.0
- **Temperature standard deviation** in m°C (signed integer) – code = value * 1000.0
- **Salinity standard deviation** in mpsu (signed integer) – code = value * 1000.0

6.5.2. Standard binary format

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The recording structures are as follows:

Subsurface - (SS)

struct

```
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    unsigned short int uiATemperature;
    unsigned short int uiSalinity;
}
```

tSSStd;

Subsurface data encoding (ascent)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Subsurface record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
				10



Raw data (drift) – (RW) or (DW)

```

struct
{
    unsigned long int uliDateTime;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
}
tRStdPark;

```

Raw data encoding (park phase)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
				10

Raw data (navigation) – (RW) or (DW)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
}
tRStdNav;

```

Raw data encoding (navigation phase)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
				8



Averaged data – (AM)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
}
tMStd;

```

Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
				8

Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    signed char cStdDeviationTemperature;
    signed char cStdDeviationSalinity;
}
tMECStd;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Standard deviation record				
Temperature STD	BYTE	-0.128 to +0.127°C	0.001°C	1
Salinity STD	BYTE	-0.128 to +0.127 psu	0.001 psu	1
				10



Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned short int uiMedianPressure;
    unsigned short int uiMedianTemperature;
    unsigned short int uiMedianSalinity;
}
tMMStd;

```

Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
				14

Averaged data, standard deviation and median – (AM)(SD)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    signed char cStdDeviationTemperature;
    signed char cStdDeviationSalinity;
    unsigned short int uiMedianPressure;
    unsigned short int uiMedianTemperature;
    unsigned short int uiMedianSalinity;
}
tMECStd;

```



Average + standard deviation + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Standard deviation record				
Temperature STD	BYTE	-0.128 to +0.127°C	0.001°C	1
Salinity STD	BYTE	-0.128 to +0.127 psu	0.001 psu	1
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
				16

Example:

[DESCENT]

(DW)

2018-11-08 16:35:23,4.2,17.471,35.798

2018-11-08 16:36:48,5.4,17.464,35.798

00000000	01 5b 44 45 53 43 45 4e 54 5d 28 44 57 29 cb 65	. [DESCENT] (DW) Ėe
00000010	e4 5b 00 00 12 04 c7 57 d6 8b 55 00 1e 04 c0 57	ä [. . . . ÇWÖ< U ÀW
00000020	d6 8b 56 00 29 04 b2 57 d5 8b 13 80 00 36 04 ab	Ö< V .) . *WÖ< . € . 6 . «
00000030	57 d4 8b 62 80 00 41 04 a4 57 d4 8b 61 80 00 4f	WÖ< b€ . A . »WÖ< a€ . O
00000040	04 9d 57 d4 8b 50 80 00 5d 04 8e 57 d2 8b 38 81	. WÖ< P€ .] . ŽWÖ< 8

Timestamp 1 = 1541694923 (0x5BE465CB) + 0 sec (0x0000) => 2018-11-08 16:35:23

Pressure 1 = 1042 (0x0412) => ((1042 / 10.0) - 100.0) = 4.2 dbar

Temperature 1 = 22471 (0x57C7) => ((22471 / 1000.0) - 5.0) = 17.471°C

Salinity 1 = 35798 (0xD68B) => (35798 / 1000.0) = 35.798 psu

Timestamp 2 = Timestamp 1 + 85 sec (0x0055) => 2018-11-08 16:36:48

...



6.5.3. Extended binary format

Advantages of this format compared to the standard version:

- Improved resolution of recordings (pressure and temperature)

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The resolution extension for pressure and temperature measurements is encoded on a shared byte using masks, as follows:

- Pressure extension (0.01 dbar) = value & 0xF0
- Temperature extension (0.1 m°C) = value & 0x0F

The available recording structures are as follows:

Subsurface – (SS)

struct

```
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    unsigned short int uiTemperature;
    unsigned short int uiSalinity;
    unsigned char ucPTExtra;
}
```

tSSExt;

Subsurface data encoding (ascent)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Subsurface record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
				11



Raw data (park) – (RW) or (DW)

```

struct
{
    unsigned long int uliDateTime;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned char ucAveragePTExtra;
}
tRExtPark;

```

Raw data encoding (park phase)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
				11

Raw data (navigation) – (RW) or (DW)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned char ucAveragePTExtra;
}
tRExtNav;

```

Raw data encoding (navigation phase)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
				9



Averaged data – (AM)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned char ucAveragePTExtra;
}
tMExt;

```

Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
				9

Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned char ucAveragePTExtra;
    signed char cStdDeviationTemperature;
    signed char cStdDeviationSalinity;
}
tMECExt;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	



Standard deviation record				
Temperature STD	BYTE	-0.128 to +0.127°C	0.001°C	1
Salinity STD	BYTE	-0.128 to +0.127 psu	0.001 psu	1
				11

Averaged data and median – (AM)(MD)

struct

```

{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned char ucAveragePTExtra;
    unsigned short int uiMedianPressure;
    unsigned short int uiMedianTemperature;
    unsigned short int uiMedianSalinity;
    unsigned char ucMedianPTExtra;
}
tMMExt;
```

Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
				16



Averaged data, standard deviation and median – (AM)(SD)(MD)

struct

```

{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageTemperature;
    unsigned short int uiAverageSalinity;
    unsigned char ucAveragePTExtra;
    signed char cStdDeviationTemperature;
    signed char cStdDeviationSalinity;
    unsigned short int uiMedianPressure;
    unsigned short int uiMedianTemperature;
    unsigned short int uiMedianSalinity;
    unsigned char ucMedianPTExtra;
}
tMECMEExt;

```

Average + standard deviation + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
Standard deviation record				
Temperature STD	BYTE	-0.128 to +0.127°C	0.001°C	1
Salinity STD	BYTE	-0.128 to +0.127 psu	0.001 psu	1
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Pressure extra	BYTE_H	0.00 to 0.09 dbar	0.01 dbar	1
Temperature extra	BYTE_L	0.0000 to 0.0009°C	0.0001°C	
				18



Example:

[DESCENT]

(DW)

2018-11-08 16:35:23,4.23,17.4716,35.798

2018-11-08 16:36:48,5.44,17.4645,35.798

00000000	01	5b	44	45	53	43	45	4e	54	5d	28	44	57	29	cb	65	[DESCENT] (DW) Ėe
00000010	e4	5b	00	00	12	04	c7	57	d6	8b	36	55	00	1e	04	c0	ä[....ÇWÖ<6U...À
00000020	57	d6	8b	45	56	00	29	04	b2	57	d5	8b	13	80	00	36	WÖ<EV.) . *WÖ< .€ .6
00000030	04	ab	57	d4	8b	62	80	00	41	04	a4	57	d4	8b	61	80	.«WÖ<b€.A. »WÖ<a€
00000040	00	4f	04	9d	57	d4	8b	50	80	00	5d	04	8e	57	d2	8b	.O. WÖ<P€.] .ŽWÖ<
00000050	38	81	00	68	04	87	57	d1	8b	67	aa	00	74	04	80	57	8 .h. #WŃ<g² .t.€W

Timestamp 1 = 1541694923 (0x5BE465CB) + 0 sec (0x0000) => 2018-11-08 16:35:23

Pressure 1 = 1042 (0x0412) + 0.03 dbar (0x3_) => ((1042 / 10.0) – 100.0) + 0.03 = 4.23 dbar

Temperature 1 = 22471 (0x57C7) + 0.0006°C (0x_6) => ((22471 / 1000.0) – 5.0) + 0.006 = 17.4716°C

Salinity 1 = 35798 (0xD68B) => (35798 / 1000.0) = 35.798 psu

Timestamp 2 = Timestamp 1 + 85 sec (0x0055) => 2018-11-08 16:36:48

...



6.6.D0 sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Phase** in degrees (float) – code = value
- **Temperature** in m°C and +5.0°C offset (unsigned integer) – code = (value + 5.0) * 1000.0
- **Phase standard deviation** in degrees (signed integer) – code = value * 1000.0
- **Temperature standard deviation** in m°C (signed integer) – code = value * 1000.0

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

struct

```
{
  unsigned long int uliDateTime;
  unsigned short int uiPressure;
  float fC1RPhase;
  float fC2RPhase;
  unsigned short int uiTemperature;
}
```

tRPark;

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
				16



Raw data (navigation) – (RW) or (DW)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiPressure;
    float fC1RPhase;
    float fC2RPhase;
    unsigned short int uiTemperature;
}
tRNav;

```

Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
				14

Averaged data – (AM)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    float fAverageC1RPhase;
    float fAverageC2RPhase;
    unsigned short int uiAverageTemperature;
}
tM;

```

Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
				14



Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    float fAverageC1RPhase;
    float fAverageC2RPhase;
    unsigned short int uiAverageTemperature;
    signed short int iStdDeviationC1RPhase;
    signed short int iStdDeviationC2RPhase;
    signed char cStdDeviationTemperature;
}
tMEC;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Standard deviation record				
C1 Phase	SHORTINT	-32.768 to +32.767 deg	0.001 deg	2
C2 Phase	SHORTINT	-32.768 to +32.767 deg	0.001 deg	2
Temperature	BYTE	-0.128 to +0.127°C	0.001°C	1
				19

Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    float fAverageC1RPhase;
    float fAverageC2RPhase;
    unsigned short int uiAverageTemperature;
    unsigned short int uiMedianPressure;
    float fMedianC1RPhase;
    float fMedianC2RPhase;
    unsigned short int uiMedianTemperature;
}
tMM;

```



Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
				26

Averaged data, standard deviation and median – (AM)(SD)(MD)

struct

```
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    float fAverageC1RPhase;
    float fAverageC2RPhase;
    unsigned short int uiAverageTemperature;
    signed short int iStdDeviationC1RPhase;
    signed short int iStdDeviationC2RPhase;
    signed char cStdDeviationTemperature;
    unsigned short int uiMedianPressure;
    float fMedianC1RPhase;
    float fMedianC2RPhase;
    unsigned short int uiMedianTemperature;
}
```

tMECM;



Average + standard deviation + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Standard deviation record				
C1 Phase	SHORTINT	-32.768 to +32.767 deg	0.001 deg	2
C2 Phase	SHORTINT	-32.768 to +32.767 deg	0.001 deg	2
Temperature	BYTE	-0.128 to +0.127°C	0.001°C	1
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
C1 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
C2 Phase	FLOAT	-90.0 to +90.0 deg	IEEE float	4
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
				31

Example:

[DESCENT]

(DW)

2018-11-08 16:13:01,3.7,38.181,8.958,25.66

2018-11-08 16:13:10,5.2,38.165,8.955,25.65

00000000	03 5b 44 45 53 43 45 4e 54 5d 28 44 57 29 8d 60	[DESCENT] (DW) `
00000010	e4 5b 00 00 0d 04 58 b9 18 42 f8 53 0f 41 c4 77	ä[...X¹.BøS.AÄw
00000020	09 00 1c 04 f6 a8 18 42 ae 47 0f 41 bd 77 04 00ö".BøG.Akw..
00000030	27 04 08 ac 18 42 f8 53 0f 41 ba 77 06 00 35 04	'...-.BøS.A°w..5.
00000040	27 b1 18 42 42 60 0f 41 b7 77 06 00 40 04 39 b4	'±.BB`.A-w..@.9'
00000050	18 42 8b 6c 0f 41 b5 77 06 00 4e 04 58 b9 18 42	.B<1.Apw..N.X¹.B

Timestamp 1 = 1541693581 (0x5BE4608D) + 0 sec (0x0000) => 2018-11-08 16:13:01

Pressure 1 = 1037 (0x040D) => ((1037 / 10.0) - 100.0) = 3.7 dbar

[...]

Temperature 1 = 30660 (0x77C4) => ((30660 / 1000.0) - 5.0) = 25.66°C

Timestamp 2 = Timestamp 1 + 9 sec (0x0009) => 2018-11-08 16:13:10

...



6.7.OCR sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Channel count** (unsigned long int) – code = value
- **Channel count standard deviation** (signed long int) – code = value

Data record can be composed of counts from 4, 7 or 14 channels (depending on the sensor's configuration).

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

struct

```
{
  unsigned long int uliDateTime;
  unsigned short int uiPressure;
  unsigned long int uliCountChannel1;
  unsigned long int uliCountChannel2;
  [...]
  unsigned long int uliCountChannelN;
}
```

tRPark;

4 channels - Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
				22



Raw data (navigation) – (RW) or (DW)

struct

```

{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiPressure;
  unsigned long int uliCountChannel1;
  unsigned long int uliCountChannel2;
  [...]
  unsigned long int uliCountChannelN;
}
tRNav;

```

4 channels - Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
				20

Averaged data – (AM)

struct

```

{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiAveragePressure;
  unsigned long int uliAverageCountChannel1;
  unsigned long int uliAverageCountChannel2;
  [...]
  unsigned long int uliAverageCountChannelN;
}
tM;

```

4 channels - Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
				20



Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned long int uliAverageCountChannel1;
    unsigned long int uliAverageCountChannel2;
    [...]
    unsigned long int uliAverageCountChannelN;
    signed long int liStdDeviationCountChannel1;
    signed long int liStdDeviationCountChannel2;
    [...]
    signed long int liStdDeviationCountChannelN;
}
tMEC;

```

4 channels - Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
Standard deviation record				
Channel 1 count	LONGINT	-2147483648 to +2147483647	1	4
Channel 2 count	LONGINT	-2147483648 to +2147483647	1	4
Channel 3 count	LONGINT	-2147483648 to +2147483647	1	4
Channel 4 count	LONGINT	-2147483648 to +2147483647	1	4
				34

Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned long int uliAverageCountChannel1;
    unsigned long int uliAverageCountChannel2;
    [...]
    unsigned long int uliAverageCountChannelN;
    unsigned short int uiMedianPressure;
    unsigned long int uliMedianCountChannel1;
    unsigned long int uliMedianCountChannel2;
    [...]
    unsigned long int uliMedianCountChannelN;
}
tMM;

```



4 channels - Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
				38

Averaged data, standard deviation and median – (AM)(SD)(MD)

struct

```
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned long int uliAverageCountChannel1;
    unsigned long int uliAverageCountChannel2;
    [...]
    unsigned long int uliAverageCountChannelN;
    signed long int liStdDeviationCountChannel1;
    signed long int liStdDeviationCountChannel2;
    [...]
    signed long int liStdDeviationCountChannelN;
    unsigned short int uiMedianPressure;
    unsigned long int uliMedianCountChannel1;
    unsigned long int uliMedianCountChannel2;
    [...]
    unsigned long int uliMedianCountChannelN;
}
tMECM;
```



4 channels - Average + standard deviation + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
Standard deviation record				
Channel 1 count	LONGINT	-2147483648 to +2147483647	1	4
Channel 2 count	LONGINT	-2147483648 to +2147483647	1	4
Channel 3 count	LONGINT	-2147483648 to +2147483647	1	4
Channel 4 count	LONGINT	-2147483648 to +2147483647	1	4
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	LONGINT	0 to +4294967295	1	4
Channel 2 count	LONGINT	0 to +4294967295	1	4
Channel 3 count	LONGINT	0 to +4294967295	1	4
Channel 4 count	LONGINT	0 to +4294967295	1	4
				54

Example:

[DESCENT]

(DW)

2018-11-08 16:13:01,3.7,2147299913,2147399789,2147931867,2147056502

2018-11-08 16:13:07,4.2,2147302930,2147406436,2147929755,2147059858

00000000	04 5b 44 45 53 43 45 4e 54 5d 28 44 57 29 8d 60	[DESCENT] (DW) `
00000010	e4 5b 00 00 0d 04 49 32 fd 7f 6d b8 fe 7f db d6	ä[...I2ý m,þ ÛÖ
00000020	06 80 76 7b f9 7f 06 00 12 04 12 3e fd 7f 64 d2	.év{ù>ý dò
00000030	fe 7f 9b ce 06 80 92 88 f9 7f 02 00 19 04 76 38	p >î.é' ^ùv8
00000040	fd 7f c9 c9 fe 7f 09 d0 06 80 db 71 f9 7f 02 00	ý ÉÉþ .Ð.éÛqù ..

Timestamp 1 = 1541693581 (0x5BE4608D) + 0 sec (0x0000) => 2018-11-08 16:13:01

Pressure 1 = 1037 (0x040D) => $((1037 / 10.0) - 100.0) = 3.7$ dbar

Count 1 = 2147299913 (0x7FFD3249)

[...]

Timestamp 2 = Timestamp 1 + 6 sec (0x0006) => 2018-11-08 16:13:07

...



6.8.ECO sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Channel count** (signed short int) – code = value
- **Channel count standard deviation** (signed char) – code = value

Data record can be composed of counts from 1 to 3 channels (depending on the sensor's configuration).

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    signed short int iCountChannel1;
    [...]
    signed short int iCountChannelN;
}
tRPark;
```

3 channels - Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	SHORTINT	-64 to +4130	1	2
Channel 2 count	SHORTINT	-64 to +4130	1	2
Channel 3 count	SHORTINT	-64 to +4130	1	2
				12



Raw data (navigation) – (RW) or (DW)

```

struct
{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiPressure;
  signed short int iCountChannel1;
  [...]
  signed short int iCountChannelN;
}
tRNav;

```

3 channels - Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	-	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	SHORTINT	-64 to +4130	1	2
Channel 2 count	SHORTINT	-64 to +4130	1	2
Channel 3 count	SHORTINT	-64 to +4130	1	2
				10

Averaged data – (AM)

```

struct
{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiAveragePressure;
  signed short int iAverageCountChannel1;
  [...]
  signed short int iAverageCountChannelN;
}
tM;

```

3 channels - Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	SHORTINT	-64 to +4130	1	2
Channel 2 count	SHORTINT	-64 to +4130	1	2
Channel 3 count	SHORTINT	-64 to +4130	1	2
				10



Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed short int iAverageCountChannel1;
    [...]
    signed short int iAverageCountChannelN;
    signed char cStdDeviationCountChannel1;
    [...]
    signed char cStdDeviationCountChannelN;
}
tMEC;

```

3 channels - Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	SHORTINT	-64 to +4130	1	2
Channel 2 count	SHORTINT	-64 to +4130	1	2
Channel 3 count	SHORTINT	-64 to +4130	1	2
Standard deviation record				
Channel 1 count	BYTE	-128 to +127	1	1
Channel 2 count	BYTE	-128 to +127	1	1
Channel 3 count	BYTE	-128 to +127	1	1
				13

Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed short int iAverageCountChannel1;
    [...]
    signed short int iAverageCountChannelN;
    unsigned short int uiMedianPressure;
    signed short int iMedianCountChannel1;
    [...]
    signed short int iMedianCountChannelN;
}
tMM;

```



3 channels - Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	SHORTINT	-64 to +4130	1	2
Channel 2 count	SHORTINT	-64 to +4130	1	2
Channel 3 count	SHORTINT	-64 to +4130	1	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	SHORTINT	-64 to +4130	1	2
Channel 2 count	SHORTINT	-64 to +4130	1	2
Channel 3 count	SHORTINT	-64 to +4130	1	2
				18

Averaged data, standard deviation and median – (AM)(SD)(MD)

struct

```
{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiAveragePressure;
  signed short int iAverageCountChannel1;
  [...]
  signed short int iAverageCountChannelN;
  signed char cStdDeviationCountChannel1;
  [...]
  signed char cStdDeviationCountChannelN;
  unsigned short int uiMedianPressure;
  signed short int iMedianCountChannel1;
  [...]
  signed short int iMedianCountChannelN;
}
```

tMECM;



21

2018-11-08 16:13:07,4.2,4130,4130,4130

[DESCENT] (DW) `

ä [. . . " " " . . .

" " " . . . " " "

. . . " " " # "

" " * " " "

...

6.9.ECOv2 sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Channel count** (unsigned short int) – code = value
- **Channel count standard deviation** (signed int) – code = value

Data record can be composed of counts from 1 to 4 channels (depending on the sensor's configuration).

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    unsigned short int uiCountChannel1;
    [...]
    unsigned short int uiCountChannelN;
}
tRPark;
```

3 channels - Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
				12



Raw data (navigation) – (RW) or (DW)

```

struct
{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiPressure;
  unsigned short int uiCountChannel1;
  [...]
  unsigned short int uiCountChannelN;
}
tRNav;

```

3 channels - Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	-	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
				10

Averaged data – (AM)

```

struct
{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiAveragePressure;
  unsigned short int uiAverageCountChannel1;
  [...]
  unsigned short int uiAverageCountChannelN;
}
tM;

```

3 channels - Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
				10



Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageCountChannel1;
    [...]
    unsigned short int uiAverageCountChannelN;
    signed short int iStdDeviationCountChannel1;
    [...]
    signed short int iStdDeviationCountChannelN;
}
tMEC;

```

3 channels - Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
Standard deviation record				
Channel 1 count	SHORTINT	-32768 to +32767	1	2
Channel 2 count	SHORTINT	-32768 to +32767	1	2
Channel 3 count	SHORTINT	-32768 to +32767	1	2
				16

Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    unsigned short int uiAverageCountChannel1;
    [...]
    unsigned short int uiAverageCountChannelN;
    unsigned short int uiMedianPressure;
    unsigned short int uiMedianCountChannel1;
    [...]
    unsigned short int uiMedianCountChannelN;
}
tMM;

```



3 channels - Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
				18

Averaged data, standard deviation and median – (AM)(SD)(MD)

struct

```
{
  unsigned short int uiDateTimeDelta;
  unsigned short int uiAveragePressure;
  unsigned short int uiAverageCountChannel1;
  [...]
  unsigned short int uiAverageCountChannelN;
  signed short int iStdDeviationCountChannel1;
  [...]
  signed short int iStdDeviationCountChannelN;
  unsigned short int uiMedianPressure;
  unsigned short int uiMedianCountChannel1;
  [...]
  unsigned short int uiMedianCountChannelN;
}
```

tMECM;



3 channels - Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
Standard deviation record				
Channel 1 count	SHORTINT	-32768 to +32767	1	2
Channel 2 count	SHORTINT	-32768 to +32767	1	2
Channel 3 count	SHORTINT	-32768 to +32767	1	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Channel 1 count	USHORTINT	0 to +65000	1	2
Channel 2 count	USHORTINT	0 to +65000	1	2
Channel 3 count	USHORTINT	0 to +65000	1	2
				24



6.10. CROVER sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Corrected signal raw count** (signed short int) – code = value
- **Corrected signal raw count standard deviation** (signed short int) – code = value

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    signed short int iCorrectedSignalRawCount;
}
tRPark;
```

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
				8

Raw data (navigation) – (RW) or (DW)

```
struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiPressure;
    signed short int iCorrectedSignalRawCount;
}
tRNav;
```



Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
				6

Averaged data – (AM)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed short int iAverageCorrectedSignalRawCount;
}
tM;

```

Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
				6

Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed short int iAverageCorrectedSignalRawCount;
    signed short int iStdDeviationCorrectedSignalRawCount;
}
tMEC;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
Standard deviation record				
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
				8



Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed short int iAverageCorrectedSignalRawCount;
    unsigned short int uiMedianPressure;
    signed short int iMedianCorrectedSignalRawCount;
}
tMM;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	FLOAT	-32768 to +32767	1	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
				10

Averaged data, standard deviation and median – (AM)(SD)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    float fAverageBeamAttenuation;
    signed short int iStdDeviationCorrectedSignalRawCount;
    unsigned short int uiMedianPressure;
    signed short int iMedianCorrectedSignalRawCount;
}
tMECM;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
Standard deviation record				
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Corr Sig Raw	SHORTINT	-32768 to +32767	1	2
				12



6.11. SBEPH sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Voltage** in μV (signed long int) – code = (value * 1000000.0)
- **Voltage standard deviation** in μV (signed char) – code = (value * 1000000.0)

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    signed long int liRefVoltage;
}
tRPark;
```

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μV	1 μV	4
				10

Raw data (navigation) – (RW) or (DW)

```
struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiPressure;
    signed long int liRefVoltage;
}
tRNav;
```



Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
				8

Averaged data – (AM)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed long int liAverageRefVoltage;
}
tM;

```

Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
				8

Averaged data and standard deviation – (AM)(SD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed long int liAverageRefVoltage;
    signed short int iStdDeviationRefVoltage;
}
tMEC;

```

Average + standard deviation data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
Standard deviation record				
Voltage	SHORTINT	-32768 to +32767 μ V	1 μ V	2
				10



Averaged data and median – (AM)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed long int liAverageRefVoltage;
    unsigned short int uiMedianPressure;
    signed long int liMedianRefVoltage;
}
tMM;

```

Average + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
				14

Averaged data, standard deviation and median – (AM)(SD)(MD)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAveragePressure;
    signed long int liAverageRefVoltage;
    signed short int iStdDeviationRefVoltage;
    unsigned short int uiMedianPressure;
    signed long int liMedianRefVoltage;
}
tMECM;

```

Average + standard deviation + median data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535 sec	1 sec	2
Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
Standard deviation record				
Voltage	SHORTINT	-32768 to +32767 μ V	1 μ V	2
Median record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Voltage	LONGINT	-2500000 to +2500000 μ V	1 μ V	4
				16



Example:

[DESCENT]

(AM)

2018-11-08 16:13:00,3.7,0.335851

2018-11-08 16:13:07,4.3,0.472219

00000000	0b	5b	44	45	53	43	45	4e	54	5d	28	41	4d	29	8c	60	[DESCENT] (AM)Æ`
00000010	e4	5b	00	00	0d	04	eb	1f	05	00	07	00	13	04	9b	34	ä[....ë.....>4
00000020	07	00	04	00	1f	04	38	51	07	00	04	00	2a	04	26	5b8Q....*.&[
00000030	07	00	04	00	33	04	a8	5a	07	00	05	00	3c	04	d9	593."Z....<.ÛY
00000040	07	00	05	00	46	04	33	56	07	00	04	00	50	04	6a	4aF.3V....P.jJ

Timestamp 1 = 1541693580 (0x5BE4608C) + 0 sec (0x0000) => 2018-11-08 16:13:00

Pressure 1 = 1037 (0x040D) => $((1037 / 10.0) - 100.0) = 3.7$ dbarVoltage = 335851 (0x00051FEB) => $(335851 / 100000.0) = 0.335851$ V

Timestamp 2 = Timestamp 1 + 7 sec (0x0007) => 2018-11-08 16:13:07

...



6.12. SUNA sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Temperature** in m°C and +5.0°C offset (unsigned integer) – code = $(\text{value} + 5.0) * 1000.0$
- **Salinity** in mpsu (unsigned integer) – code = $\text{value} * 1000.0$
- **Humidity** in % (unsigned char) – code = $(\text{value} * 2.0)$
- **Dark mean** (unsigned short int) – code = $(\text{value} * 10.0)$
- **Dark standard deviation** (signed short int) – code = $(\text{value} * 100.0)$
- **Nitrate concentration** in μM (float) – code = value
- **Absorbance fit residual RMS** (float) – code = value
- **Output spectrum** (unsigned short int) – code = value

Data record can be composed of output spectrum from 45 or 90 channels (depending on the sensor's configuration).

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

The available recording structures are as follows:

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    struct
    {
        struct
        {
            unsigned short int uiPressure;
            unsigned short int uiTemperature;
            unsigned short int uiSalinity;
        }
        tCTD;
        unsigned short int uiInternalTemperature;
        unsigned short int uiSpectrometerTemperature;
        unsigned char ucInternalRelativeHumidity;
        struct
        {
            unsigned short int uiMean;
            signed short int iStandardDeviation;
        }
        tDarkSpectrum;
    }
    struct
```



```

{
  float fNitrate;
}
tSensor;
float fAbsorbanceFitResiduals;
struct
{
  unsigned short int tuiSpectrum[ N ];
}
tOutput;
}
tAPFData;
}
tRPark;

```

45 outputs spectrum - Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Temperature spec.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Relative humidity	BYTE	0.0 to +100% RH	0.5% RH	1
Dark M	SHORTINT	0.0 to +6553.5	0.1	2
Dark SD	SHORTINT	-327.68 to +327.67	0.01	2
Nitrate	FLOAT	IEEE float	IEEE float	4
Absorbance	FLOAT	IEEE float	IEEE float	4
Output 1	SHORTINT	0 to +65535	1	2
...	-	-	-	-
Output 45	SHORTINT	0 to +65535	1	2
				117



Raw data (navigation) – (RW) or (DW)

```

struct
{
  unsigned short int uiDateTimeDelta;
  struct
  {
    struct
    {
      unsigned short int uiPressure;
      unsigned short int uiTemperature;
      unsigned short int uiSalinity;
    }
    tCTD;
    unsigned short int uiInternalTemperature;
    unsigned short int uiSpectrometerTemperature;
    unsigned char ucInternalRelativeHumidity;
    struct
    {
      unsigned short int uiMean;
      signed short int iStandardDeviation;
    }
    tDarkSpectrum;
    struct
    {
      float fNitrate;
    }
    tSensor;
    float fAbsorbanceFitResiduals;
    struct
    {
      unsigned short int tuiSpectrum[ N ];
    }
    tOutput;
  }
  tAPFData;
}
tRNav;

```

45 outputs spectrum - Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Salinity	SHORTINT	0.0 to +50.0 psu	0.001 psu	2
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Temperature spec.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Relative humidity	BYTE	0.0 to +100% RH	0.5% RH	1
Dark M	SHORTINT	0.0 to +6553.5	0.1	2



Dark SD	SHORTINT	-327.68 to +327.67	0.01	2
Nitrate	FLOAT	IEEE float	IEEE float	4
Absorbance	FLOAT	IEEE float	IEEE float	4
Output 1	SHORTINT	0 to +65535	1	2
...	-	-	-	-
Output 45	SHORTINT	0 to +65535	1	2
				115

Example:

[DESCENT]

(DW)

2018-11-21 10:38:51,1.5,17.486,[...],8.0,[...]

2018-11-21 10:39:20,9.2,[...]

00000000	0c 5b 44 45 53 43 45 4e	54 5d 28 44 57 29 bb 35	[DESCENT] (DW) »5
00000010	f5 5b 00 00 f7 03 d6 57	d7 8b ba 68 42 68 10 06	õ[...÷.ÖW×× °hBh..
00000020	1d 84 03 1f 85 5c c2 94	87 85 3c c0 9c a7 a7 ea\Â"±...<ÀæSSé
00000030	b3 8c c0 fd cc 95 d8 2a	e2 00 e9 2c ec 72 eb 7c	'CAYİ•0*â.é,irē
00000040	e7 87 e0 9b d7 04 ce 98	c4 f1 bb 74 b4 a2 ae 11	ç+à>×.Î~Ãñ»t'©.
00000050	aa 08 a7 62 a5 f4 a4 b6	a5 54 a7 04 aa 91 ad 45	².\$bYô×¶YTS.²`-E
00000060	b2 a9 b7 d6 bd 95 c4 cc	cb 40 d3 74 da d8 e0 40	*@·Ö×•ÄIÊ@ÓtÚ@à@
00000070	e6 bd e9 00 eb e8 e9 53	e6 4c e0 b1 d8 39 cf 00	æ%é.ëëéSæLâ±09İ.
00000080	00 00 00 00 00 1d 00 44	04 a4 57 d4 8b 00 69 7eD.×WÔ<.i~
00000090	68 10 06 1d 20 03 3d 0a	5c c2 85 b1 85 3c 8c 9c	h... .=\Â..±...<@æ
000000a0	83 a7 a7 b3 4d c0 bc cc	42 d8 e0 e1 be e8 e1 eb	fSS'MÀ×IB0ää×eäé
000000b0	22 eb 25 e7 38 e0 5e d7	c6 cd 56 c4 ca bb 44 b4	"ëç8â×EÍVÄÊ»D'
000000c0	68 ae f7 a9 e2 a6 4a a5	b9 a4 82 a5 20 a7 d4 a9	hø=øâ!JY¹×,¥ \$Ôø
000000d0	69 ad 04 b2 67 b7 9c bd	6e c4 86 cb 0a d3 4a da	i-.²g·æ×nÄ+Ê.ÓJÚ
000000e0	8a e0 ec e5 74 e9 b4 ea	a1 e9 21 e6 30 e0 88 d8	Šaiâté'ê;æ!æ0â^0
000000f0	21 cf 00 00 00 00 00 00	1e 00 7e 04 79 57 d1 8b	!İ.....~.yWÑ<

Timestamp 1 = 1542796731 (0x5BE435BB) + 0 sec (0x0000) => 2018-11-21 10:38:51

Pressure 1 = 1015 (0x03F7) => ((1015 / 10.0) - 100.0) = 1.5 dbar

Temperature 1 = 22486 (0x57D6) => ((22486 / 1000.0) - 5.0) = 17.486°C

[...]

Humidity 1 = 16 (0x10) => (16 / 2.0) = 8.0 % RH

Timestamp 2 = Timestamp 1 + 29 sec (0x001D) => 2018-11-21 10:39:20

...



6.13. UVP6 sensor (LPM)

Data is encoded as follows:

- **Timestamping** of each record
- **Number of images averaged** in a record (unsigned char) – code = value
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = (value + 100.0) * 10.0
- **Temperature int.** in m°C and +5.0°C offset (unsigned integer) – code = (value + 5.0) * 1000.0
- **Number of particles per size class** (float) – code = value
- **Mean grey level per size class** (unsigned char) – code = value

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

Raw data (park) – (RW) or (DW)

struct

```
{
    unsigned long int uliDateTime;
    unsigned char ucAverageNumber;
    unsigned short int uiPressure;
    unsigned short int uiInternalTemperature;
    float tfObjectCount[ 18 ];
    unsigned char tucGreyLevel[ 18 ];
}
```

tRPark;

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Image number	BYTE	1 to +255	1	1
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Particles 1 count	FLOAT	IEEE float	IEEE float	4
...	-	-	-	16*4
Particles 18 count	FLOAT	IEEE float	IEEE float	4
Grey level 1	BYTE	0 to 255	1	1
...	-	-	-	16*1
Grey level 18	BYTE	0 to 255	1	1
				99



Raw data (navigation) – (RW) or (DW)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiPressure;
    unsigned short int uiInternalTemperature;
    float tfObjectCount[ 18 ];
    unsigned char tucGreyLevel[ 18 ];
}
tRNav;

```

Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Particles 1 count	FLOAT	IEEE float	IEEE float	4
...	-	-	-	16*4
Particles 18 count	FLOAT	IEEE float	IEEE float	4
Grey level 1	BYTE	0 to 255	1	1
...	-	-	-	16*1
Grey level 18	BYTE	0 to 255	1	1
				96

Averaged data – (AM)

```

struct
{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiAverageNumber;
    unsigned short int uiAveragePressure;
    unsigned short int uiInternalTemperature;
    float tfObjectCount[ 18 ];
    unsigned char tucGreyLevel[ 18 ];
}
tM;

```



Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535	1 sec	2
Average record				
Image number	SHORTINT	1 to +65535	1	2
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Particles 1 count	FLOAT	IEEE float	IEEE float	4
...	-	-	-	16*4
Particles 18 count	FLOAT	IEEE float	IEEE float	4
Grey level 1	BYTE	0 to 255	1	1
...	-	-	-	16*1
Grey level 18	BYTE	0 to 255	1	1
				98

Example:

[DESCENT]

(DW)

2019-09-23 15:29:34,0,0.00,24.31,27682.0000,2576.0000,0.0000,510.0000,[...],2,2,0,2,2,2,0,[...]

00000000	0e 5b 44 45 53 43 45 4e 54 5d 28 44 57 29 de e4	[DESCENT] (DW) Pă
00000010	88 5d 00 00 e8 03 7e 72 00 44 d8 46 00 00 21 45	^] ..è.~r.DØF...!E
00000020	00 00 00 00 00 00 ff 43 00 00 e0 41 00 00 c0 40ÿC..âA..À@
00000030	00 00 80 3f 00 00 00 00 00 00 00 00 00 00 00 00	..€?.....
00000040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000060	02 02 00 02 02 02 02 00 00 00 00 00 00 00 00 00

Timestamp 1 = 1569252574 (0x5D88E4DE) + 0 sec (0x0000) => 2019-09-23 15:29:34

Pressure 1 = 1015 (0x03E8) => ((1000 / 10.0) - 100.0) = 0.0 dbar

Temperature 1 = 29310 (0x727E) => ((29310 / 1000.0) - 5.0) = 24.310°C

Particle 1/18 1 = 27682.0 (0x46D84400)

[...]

Grey level 1 = 2 (0x02)



6.14. UVP6 sensor (BLACK)

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Internal temperature** in m°C and +5.0°C offset (unsigned integer) – code = $(\text{value} + 5.0) * 1000.0$
- **Number of particles per size class** (unsigned integer) – code = value

All records are timestamped in absolute value.

Raw data – (RW)

struct

```
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    unsigned short int uiInternalTemperature;
    unsigned short int tuiObjectCount[ 5 ];
}
```

tR;

Raw data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Particles 1 count	SHORTINT	0 to +65535	1	2
...	-	-	-	3*2
Particles 5 count	SHORTINT	0 to +65535	1	2
				18



Example:

[DESCENT]

(RW)

2019-04-30 14:56:26,3.3,1,31.94,8504,1001,0,511,88

2019-04-30 14:56:57,10.9,1,31.94,12702,1226,0,591,128

00000000	11	5b	44	45	53	43	45	4e	54	5d	28	52	57	29	1a	62	[DESCENT] (RW).b
00000010	c8	5c	09	04	01	4c	90	38	21	e9	03	00	00	ff	01	58	È\...L 8!é...ÿ.X
00000020	00	39	62	c8	5c	55	04	01	4c	90	9e	31	ca	04	00	00	.9bÈ\U...L ž1Ê...
00000030	4f	02	80	00	69	62	c8	5c	b5	04	01	88	90	11	3f	0d	O.€.ibÈ\p...^ .?.
00000040	06	00	00	5e	02	67	00	5b	50	41	52	4b	5d	78	62	c8	...^..g.[PARK]xbÈ
00000050	5c	cb	04	01	88	90	c8	2f	c1	04	00	00	2a	02	73	00	\È...^ È/Á...*.s.

Timestamp 1 = 1556636188 (0x5CC8621A) => 2019-04-30 14:56:26
 Pressure 1 = 1033 (0x0409) => $((1033 / 10.0) - 100.0) = 3.3$ dbar
 Temperature 1 = 36940 (0x904C) => $((36940 / 1000.0) - 5.0) = 31.94^{\circ}\text{C}$
 Object count class_1 1 = 8504 (0x2138)
 [...]
 Timestamp 2 = 1556636217 (0x5CC86239) => 2019-04-30 14:56:57
 Pressure 2 = 1109 (0x0455) => $((1109 / 10.0) - 100.0) = 10.9$ dbar
 ...



6.15. EXTTRIG sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$

All records are timestamped in absolute value. So, there are no processing type identification strings inserted in the data file.

Raw data – (RW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
}
tR;
```

Raw data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
				6

Example:

[DESCENT]

(RW)

2020-01-20 14:08:53,0.0

2020-01-20 14:08:55,0.0

00000000	16 5b 44 45 53 43 45 4e 54 5d 75 b4 25 5e e8 03	[DESCENT]u's^è. w's^è.z's^è.} 's^ è.€'s^è.f's^è.t' s^è.‰'s^è.œ's^i.
00000010	77 b4 25 5e e8 03 7a b4 25 5e e8 03 7d b4 25 5e	
00000020	e8 03 80 b4 25 5e e8 03 83 b4 25 5e e9 03 86 b4	
00000030	25 5e ea 03 89 b4 25 5e eb 03 8c b4 25 5e ec 03	

Timestamp 1 = 1579529333 (0x5E25B475) => 2020-01-20 14:08:53
 Pressure 1 = 1000 (0x03E8) => $((1000 / 10.0) - 100.0) = 0.0$ dbar
 Timestamp 2 = 1579529335 (0x5E25B477) => 2020-01-20 14:08:55
 Pressure 2 = 1000 (0x03E8) => $((1000 / 10.0) - 100.0) = 0.0$ dbar
 ...



6.16. RAMSES sensor

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Integration time** (unsigned integer) – code = value
- **Depth** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 20.0$
- **Inclination** in cdegree (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Dark average** (unsigned integer) – code = value
- **Number of spectrum channels** (unsigned char) – code = value
- **Spectrum channels count** (unsigned integer) – code = value (spectrum size is not constant)

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    unsigned short int uiIntegrationTime;
    unsigned short int uiPrePressure;
    unsigned short int uiPostPressure;
    unsigned short int uiPreInclination;
    unsigned short int uiPostInclination;
    unsigned short int uiDark;
    unsigned char ucSpectrumSize;
    unsigned short int tuiSpectrum[ 245 max ];
}
tRPark;
```

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Integration time	SHORTINT	0 to +65535	1	2
Pre-Pressure	SHORTINT	-100 to +2500 dbar	0.05 dbar	2
Post-Pressure	SHORTINT	-100 to +2500 dbar	0.05 dbar	2
Pre-Inclination	SHORTINT	0 to +360°	0.01°	2
Post-Inclination	SHORTINT	0 to +360°	0.01°	2
Dark average	SHORTINT	0 to +65535	1	2
Channel number N	BYTE	0 to +245	1	1



Channel 1 count	SHORTINT	0 to +65535	1	2
...	-	-	-	(N-2)*2
Channel N count	SHORTINT	0 to +65535	1	2
				19+2N

Raw data (navigation) – (RW) or (DW)

struct

```

{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiPressure;
    unsigned short int uiIntegrationTime;
    unsigned short int uiPrePressure;
    unsigned short int uiPostPressure;
    unsigned short int uiPreInclination;
    unsigned short int uiPostInclination;
    unsigned short int uiDark;
    unsigned char ucSpectrumSize;
    unsigned short int tuiSpectrum[ 245 max ];
}

```

tRNav;

Raw data encoding (navigation)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535	1 sec	2
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Integration time	SHORTINT	0 to +65535	1	2
Pre-Pressure	SHORTINT	-100 to +2500 dbar	0.05 dbar	2
Post-Pressure	SHORTINT	-100 to +2500 dbar	0.05 dbar	2
Pre-Inclination	SHORTINT	0 to +360°	0.01°	2
Post-Inclination	SHORTINT	0 to +360°	0.01°	2
Dark average	SHORTINT	0 to +65535	1	2
Channel number N	BYTE	0 to +245	1	1
Channel 1 count	SHORTINT	0 to +65535	1	2
...	-	-	-	(N-2)*2
Channel N count	SHORTINT	0 to +65535	1	2
				17+2N



Example:

[DESCENT]

(DW)

2020-03-30 12:26:36,0.60,4096,0.95,0.95,271.27,271.28,1785,50,1758,1755,1762,...,4865

2020-03-30 12:26:38,1.50,4096,0.95,0.95,271.27,271.28,1785,50,1758,1755,1762,...,4865

00000000	17 5b 44 45 53 43 45 4e 54 5d 28 44 57 29 7c e5	[DESCENT] (DW) Å
00000010	81 5e 00 00 ee 03 00 10 e3 07 e3 07 f7 69 f8 69	^..î...ã.ã.÷iøi
00000020	f9 06 32 de 06 db 06 e2 06 e7 06 e2 06 e6 06 ec	ù.2ß.Û.â.ç.â.æ.i
00000030	06 ee 06 f0 06 fb 06 fb 06 06 07 16 07 44 07 d3	.î.8.û.û.....D.Ó
00000040	07 c2 08 f4 09 35 0b e5 0c 67 10 3d 13 e4 14 7e	.Â.ô.5.â.g.=.ä.~
00000050	14 44 13 16 12 8f 11 3f 11 57 11 4d 12 33 14 d8	.D... .?.W.M.3.Ø
00000060	16 41 18 b7 18 f3 18 89 19 86 1b f6 19 f5 17 30	.A. .ó.%+.ö.ö.0
00000070	17 cc 16 d7 16 12 16 ac 15 6d 15 e6 15 c8 19 4c	.Ï.*...-.m.æ.È.L
00000080	1d 89 18 07 14 01 13 02 00 f7 03 00 10 e3 07 e3	.%......÷...ã.ã

Timestamp 1 = 1585571196 (0x5E81E57C) => 2020-03-30 12:26:36

Pressure 1 = 1006 (0x03EE) => ((1006 / 10.0) – 100.0) = 0.6 dbar

Pre-Inclination 1 = 27128 (0x69F7) => (27127 * 100.0) = 271.28°

Channel number 1 = 50 (0x32)

[...]

Timestamp 2 = Timestamp 1 + 2 sec (0x0002) => 2020-03-30 12:26:38



6.18. OPUS sensor (LIGHT)

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Spectrum type** (char) – code = '2' for calibrated spectrum and '4' for raw light spectrum
- **Averaging** (unsigned char) – code = $\text{value} * 10$
- **Flash count** (unsigned char) – code = $\text{value} * 10$
- **Internal temperature** in m°C and +5.0°C offset (unsigned integer) – code = $(\text{value} + 5.0) * 1000.0$
- **Number of spectrum channels** (unsigned char) – code = value
- **Spectrum channels count** (unsigned integer) – code = value (spectrum size is not constant)

During the "descent" and "ascent" navigation phases, the first record of each processing area is timestamped in absolute value (reference date), then the recordings for the next records are timestamped in relative value by encoding the date in relation to the reference date (deviation in seconds).

During the "drift" phases, all records are timestamped in absolute value.

Raw data (park) – (RW) or (DW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    char cSpectrumType;
    unsigned char ucAveraging;
    unsigned char ucFlashCount;
    unsigned short int uiTemperature;
    unsigned char uiSpectrumSizeN;
    unsigned short int tuiSpectrumN[ 255 max ];
    unsigned char uiSpectrumSizeM;
    unsigned short int tuiSpectrumM[ 255 max ];
}
tRPark;
```

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Spectrum type	BYTE	'2': calibrated / '4': raw	1	1
Averaging	BYTE	0 to 25.0	0.1	1
Flash count	BYTE	0 to 25.0	0.1	1
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Channel number N	BYTE	0 to +255	1	1
Channel F1 count	SHORTINT	0 to +65535	1	2



...	-	-	-	$(N-2)*2$
Channel FN count	SHORTINT	0 to +65535	1	2
Channel number M	BYTE	0 to +255	1	1
Channel B1 count	SHORTINT	0 to +65535	1	2
...	-	-	-	$(M-2)*2$
Channel BM count	SHORTINT	0 to +65535	1	2
				13+2N+2M

Raw data (navigation) – (RW) or (DW)/Averaged data – (AM)

struct

```

{
    unsigned short int uiDateTimeDelta;
    unsigned short int uiPressure;
    char cSpectrumType;
    unsigned char ucAveraging;
    unsigned char ucFlashCount;
    unsigned short int uiTemperature;
    unsigned char uiSpectrumSizeN;
    unsigned short int tuiSpectrumN[ 255 max ];
    unsigned char uiSpectrumSizeM;
    unsigned short int tuiSpectrumM[ 255 max ];
}

```

tRNav/tM;

Raw data encoding (navigation)/ Average data encoding				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	SHORTINT	0 to +65535	1 sec	2
Raw record/Average record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Spectrum type	BYTE	'2': calibrated / '4': raw	1	1
Averaging	BYTE	0 to 25.0	0.1	1
Flash count	BYTE	0 to 25.0	0.1	1
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Channel number N	BYTE	0 to +255	1	1
Channel F1 count	SHORTINT	0 to +65535	1	2
...	-	-	-	$(N-2)*2$
Channel FN count	SHORTINT	0 to +65535	1	2
Channel number M	BYTE	0 to +255	1	1
Channel B1 count	SHORTINT	0 to +65535	1	2
...	-	-	-	$(M-2)*2$
Channel BM count	SHORTINT	0 to +65535	1	2
				11+2N+2M



Example:

[ASCENT]

(RW)

2020-03-30 12:26:36,1001.0,4,3.0,1.0,25.750,31,16515,[...],19,25358,[...]

00000000	18	5b	41	53	43	45	4e	54	5d	28	52	57	29	7c	e5	81	[ASCENT] (RW) ä
00000010	5e	00	00	02	2b	34	1e	0a	1e	78	1f	83	40	7a	3e	ca	^...+4...x.f@z>Ê
00000020	3b	87	39	25	38	41	37	14	37	89	37	94	37	23	38	01	;+9%8A7.7%7"7#8.
00000030	3a	6a	3e	ad	47	4d	57	77	72	3d	9a	80	b5	28	ac	2e	:j>-GMWwr=š€µ(¬.
00000040	92	6c	7e	43	6f	fe	67	25	69	f2	68	8f	65	af	61	e5	'l~Cobpgšioh e¬aâ
00000050	5c	8a	58	2d	56	ca	56	72	5c	13	0e	63	07	3e	bf	2f	\ŠX-VÊVr\...c.>¿/
00000060	49	38	72	2e	41	20	09	1e	9a	21	7e	2a	f7	25	6d	1e	I8r.A ..š!~*+šm.
00000070	33	17	b4	12	4c	11	31	11	7f	11	36	12	a8	12	d5	13	3.'L.1. .6.¬.Ö.
00000080	28	52	57	29	ef	07	5c	49	00	00	74	09	34	1e	0a	6b	(RW) i.\I..t.4..k

Timestamp 1 = 1585571196 (0x5E81E57C) => 2020-03-30 12:26:36
 Pressure 1 = 11010 (0x2B02) => ((11010 / 10.0) - 100.0) = 1001.0 dbar
 Spectrum type 1 = "4" (0x34)
 Averaging 1 = 30 (0x1E) => (30 / 10.0) = 3.0
 Flash count 1 = 10 (0x0A) => (10 / 10.0) = 1.0
 Temperature 1 = 30730 (0x781E) => ((30750 / 1000.0) - 5.0) = 25.750°C
 Spectrum size N 1 = 31 (0x1F)
 Channel F1 1 = 16515 (0x4083)
 [...]
 Spectrum size M 1 = 19 (0x13)
 Channel B1 1 = 25358 (0x630E)
 [...]



6.19. OPUS sensor (BLACK)

Data is encoded as follows:

- **Timestamping** of each record
- **Pressure** in cbar and +100 dbar offset (unsigned integer) – code = $(\text{value} + 100.0) * 10.0$
- **Averaging** (unsigned char) – code = $\text{value} * 10$
- **Flash count** (unsigned char) – code = $\text{value} * 10$
- **Internal temperature** in m°C and +5.0°C offset (unsigned integer) – code = $(\text{value} + 5.0) * 1000.0$
- **Dark mean** (float) – code = value
- **Dark standard deviation** (float) – code = value

All records are timestamped in absolute value.

Raw data – (RW)

```
struct
{
    unsigned long int uliDateTime;
    unsigned short int uiPressure;
    unsigned char ucAveraging;
    unsigned char ucFlashCount;
    unsigned short int uiTemperature;
    float fDarkMean;
    float fDarkStdDev;
}
tR;
```

Raw data encoding (park)				
Field	Format	Range	Resolution	Size (Byte)
Timestamp	EPOCH	-	1 sec	4
Raw record				
Pressure	SHORTINT	-100 to +2500 dbar	0.1 dbar	2
Averaging	BYTE	0 to 25.0	0.1	1
Flash count	BYTE	0 to 25.0	0.1	1
Temperature int.	SHORTINT	-5.0 to +40.0°C	0.001°C	2
Dark mean	FLOAT	IEEE float	IEEE float	4
Dark standard dev	FLOAT	IEEE float	IEEE float	4
				18



Example:

[ASCENT]

(RW)

2020-03-30 12:26:36,1001.0,3.0,1.0,25.687,23217.0,8959.0

00000000	19	5b	41	53	43	45	4e	54	5d	28	52	57	29	7c	e5	81	[. [ASCENT] (RW) å
00000010	5e	02	2b	1e	0a	df	77	16	62	b5	46	19	fc	0b	46	08	^..+...ßw.buF.ü.F.
00000020	08	5c	49	02	2b	1e	0a	86	78	c8	3f	b6	46	17	68	0c	.\I.+...+xÈ?¶F.h.
00000030	46	28	52	57	29	28	08	5c	49	6a	09	1e	0a	56	79	f8	F(RW) (. \Ij...Vyz

Timestamp 1 = 1585571196 (0x5E81E57C) => 2020-03-30 12:26:36
 Pressure 1 = 11010 (0x2B02) => ((11010 / 10.0) – 100.0) = 1001.0 dbar
 Averaging 1 = 30 (0x1E) => (30 / 10.0) = 3.0
 Flash count 1 = 10 (0x0A) => (10 / 10.0) = 1.0
 Temperature 1 = 30687 (0x77DF) => ((30687 / 1000.0) – 5.0) = 25.687°C
 Dark mean 1 = 23217.0 (0x46B56216)
 Dark std deviation 1 = 8959.0 (0x460BFC19)
 [...]



nke Instrumentation

6, rue Gutenberg
ZI de Kerandré
56700 Hennebont
FRANCE
Phone: +33 (0)2 97 36 10 12

www.nke-instrumentation.com

